FONDAMENTI DI INFORMATICA

DODICESIMA LEZIONE

Prof. Alfredo Accattatis

(accattatis@ing.uniroma2.it)

Tutor: Prof. Vittorio Calafiore

(vcalafio@gmail.com)

ESAMI

- 90/60 minuti di tempo
- Due domande a risposta aperta, max UN foglio A4 a domanda
- Tre esercizi

UNIVERSITA' DEGLI STUDI DI ROMA
"TOR VERGATA"

TOR VERGATA
UNIVERSITÀ
DEGLI STUDI
DI ROMA

Dipartimento di Ingegneria Industriale

Corso di studio in Ingegneria Meccanica/Energetica

Fondamenti di Informatica, sessione Estiva I appello

A.A. 2022/23

Nome	
Cognome	
Matricola	
Corso di studio	
Firma e data	

(1) Descrivere la differenza tra *script* e *function*; dire se le variabili definite internamente ad una *function* sono visibili all'interno di uno *script* che chiama la *function* e argomentare la risposta.

Data e Firma _____

(2) Descrivete le principali *strutture dati* definite in Matlab

Data e Firma _____

(3) Lo scorso anno uno studente ha effettuato la codifica di un algoritmo dato come prova d'esame. Non ci sono errori logici, ma ci sono degli errori sintattici/grammaticali. Quali? (Potete evidenziare errore e correzione direttamente sul codice proposto).

```
function [ R ] =  Estrai[ A, x ]

R = ();

if isempty( A )
    disp('Errore: matrice d'ingresso vuota. ');
    return
else
    ncol = size( A,2 );
    for i = 1:size( A,1 )
        trovato = false;
        for j = 1,ncol
           if A(i,j) = x
              trovato = true;
              break;
           end
        end
        if trovato
           CS = 0;
           for k = 1,ncol
              CS = CS + A( i,k );
           end
           R = [ R; A( i,: ), CS ];
        end
    end
end

end
```

Data e Firma _____

(4) Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando solo cicli *for*/*while*, istruzioni condizionali (*if*, *switch*), e se ritenuto necessario le funzioni predefinite *zeros*, *size*, *isempty*, *fprintf*, *disp*: si scriva una funzione Matlab/Octave che ha come parametri d'ingresso una matrice **A** di dimensione **m*n** e restituisce in uscita la medesima matrice con le righe invertite (l'ultimo elemento della riga diventa il primo, il penultimo il secondo eccetera). Come prima cosa, verificare che m>2 ed n>3, *altrimenti* uscire dalla function, restituire una matrice **m*n** contenente tutti zeri e stampare a video le dimensioni **m** ed **n**.

Data e Firma _____

(5) Sia dato il seguente frammento di codice Matlab/Octave (somma cumulativa):

```
v = [ 1 2 3 9 6 4 7 12 9 ]
n = length(v)
s = zeros( n );
i = 1;
while i <= n
    j = 1;
    while j <= i
        s( i ) = s( i ) + v( j );
        j = j + 1;
    end
    fprintf('Somma parziale %d indice %d \n', s( i ), j);
    i = i + 1;
end
```

Eseguendo il codice cosa appare a video?

Data e Firma _____

Sessione estiva AA 2015/16

(1) Descrivere la differenza tra codice ad alto livello, codice eseguibile e come si passa dall'uno all'altro.

(2) A cosa serve un linguaggio di programmazione? In che relazione si trova con il concetto di algoritmo?

(1) Un programmatore ha codificato un algoritmo che calcola la minima distanza dal valor medio. La distanza è definita come d = |x – xm| (valore assoluto della differenza tra un valore x e la media dei valori xm). Il programmatore ha commesso degli errori. Quali?

```
1.  A = [ 4 6 7 10 19 ];
2.  a = 0;
3.  for x == A
4.      a = a + x;
5.  end
6.  a = a / length( A );
7.  closest = 1;
8.  myMin = abs[ A(1) - a ];
9.  for i = 2 : length( A )
10.     if abs[ A( i ) - a ] < myMin
11.         closest = i;
12.         myMin = abs[ A( i ) - a ];
13.     end
14. end
```

Esercizio «chiave»

- Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando cicli for/while: data una matrice n x m (con n ed m >=1) scrivere un programma che restituisce in uscita un vettore le cui componenti sono i checksum delle singole righe. Il checksum di una sequenza numerica è la somma algebrica dei suoi valori.

```
1. clear
2. clc
3. n=input('numero delle righe ');
4. m=input('numero delle colonne ');
5. M=round(rand(n,m).*100);  % Matrice di esempio
6. disp(M);
7. a=0;
8. for i=1:n
9.     for j=1:m
10.        a = a + M(i,j);
11.    end;
12.    v( i ) = a;
13.    a = 0;
14.end;
15.disp( v )
16.fprintf( '%d\n', v );
```

```
1. function v = Calcola(M)
2.    [n m] = size(M);
3.    disp(M);
4.    a=0;
5.    for i=1:n
6.       for j=1:m
7.          a = a + M(i,j);
8.       end;
9.       v( i )=a;
10.       a = 0;
11.   end;
12.   disp( v )
13.   fprintf( '%d\n', v );
14.end
```

(1) Dato il seguente frammento di codice Matlab/Octave:

```
1.  m1 = [1; 7; 4; 5; 9; 2]
2.  m2 = [3, 1, 6, 2, 9, 8, 5];
3.  a = m1(1:2:end);
4.  b = m2(end:-3:1)
5.  P = a*b
6.  [r c] = size(P);
7.  d = r*c
8.  Ro = d/numel(P);
9.  fprintf('Il rapporto è %.2f\n', Ro);
10.
11.
```

- Eseguendo il programma cosa appare a video?

Prodotto tra matrici

$$[C]_{m \times p} = [A]_{m \times n} * [B]_{n \times p}$$

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

- Ad esempio

```
>> A=[3  8  0;
      1  2  5];
>> B=[1  2  3  1
      4  5  1  2
      0  2  3  0]
>> C = A*B
C =
      35    46    17    19
      9     22    20    5
>> D = B*A
Error using  *
Inner matrix dimensions must agree.
```

```
m1 = [1; 7; 4; 5; 9; 2]
m2 = [3, 1, 6, 2, 9, 8, 5];
a = m1(1:2:end);
b = m2(end:-3:1)
P = a*b
[r c] = size(P);
d = r*c
Ro = d/numel(P);
fprintf('Il rapporto è %.2f\n', Ro);
```

```
m1 =

     1
     7
     4
     5
     9
     2


b =

     5     2     3
%( a = 1;  4;   9;)

P =

     5     2     3
    20     8    12
    45    18    27


d =

     9

Il rapporto è 1.00
```

A.A. 2022/23 sessione straordinaria

1. Cosa si intende per compilazione? In cosa differisce dall'interpretazione? Dire se Matlab usa l'una o l'altra tecnica o entrambe.

2. Descrivere cosa si intende per variabili locali, dove (e se) vengono usate in Matlab e a che scopo.

A.A. 2022/23 sessione Estiva I appello (fila 2)

- Descrivere la differenza tra *script* e *function*; si possono usare le tecniche di debug indifferentemente in *script* e *function*? Argomentare la risposta parlando anche del *debug* in Matlab/Octave.
- A cosa serve il *teorema del campionamento*? Perché è importante in informatica?

A.A. 2022/23 sessione straordinaria

- Uno studente ha effettuato la codifica di un algoritmo dato come prova d'esame. Esso è codificato correttamente (non ci sono errori logici) ma ci sono degli errori sintattici/grammaticali. Quali? (potete evidenziare errore e correzione direttamente sul codice proposto). *Nota*: la function conta il numero di sequenze di dimensione 2 (numeri uguali consecutivi) all'interno del vettore "v".

```
1.  function ( num_seq; elem ) = sequenze( v )
2.  if length[ v ] < 2
3.      num_seq = 0;
4.      elem = ()    %vettore nullo
5.  else
6.      j = 1;
7.      num_seq = 0;
8.      for i == 1:length( v ) – 1
9.          if v( i ) = v( i + 1 )
10.             num_seq + 1 = num_seq    %incrementa num_seq
11.             elem( j ) = v( i )
12.             j = j + 1;
13.         end
14.     end
15. end
16. end
17. end
```

```
1.  function [ num_seq, elem ] = sequenze( v )
2.      if length(v) < 2
3.          num_seq = 0;
4.          elem = []; %vettore nullo
5.      else
6.          j = 1;
7.          num_seq = 0;
8.              for i = 1:length( v ) - 1
9.                  if v( i ) == v( i + 1 )
10.                      num_seq = num_seq + 1; %incrementa num_seq
11.                     elem( j ) = v( i );
12.                     j = j + 1;
13.             end
14.         end
15.     end
16. end
```

Fondamenti di Informatica, sessione Estiva I appello A.A. 2022/23

- Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando solo cicli *for*/*while*, istruzioni condizionali (*if*, *switch*), e se ritenuto necessario le funzioni predefinite *zeros, size, isempty, fprintf, disp*: si scriva una funzione Matlab/Octave che ha come parametri d'ingresso una matrice **A** di dimensione **m*n** e restituisce in uscita la medesima matrice con le righe invertite (l'ultimo elemento della riga diventa il primo, il penultimo il secondo eccetera). Come prima cosa, verificare che m>2 ed n>3, *altrimenti* uscire dalla function, restituire una matrice **m*n** contenente tutti zeri e stampare a video le dimensioni **m** ed **n**.

```
1.  function [ A ] = Esame2023( A )
2.
3.  [ r, c ] = size( A );
4.
5.  % l'esercizio richiede dimensioni precise
6.  if ( (r < 2)  || ( c < 3 ) )
7.    A = zeros(r, c);
8.    fprintf( 'Matrice con %d righe e %d colonne', r, c );
9.    return;
10. end
11.
12.
13. % Esegue inversione della riga
14. for i = 1 : r
15.   Temp = A( i,: );
16.   for j = 1 : c
17.      A( i,j ) = Temp( c - j + 1 );
18.   end
19. end
20.
21.
22. end
```

Sia dato il seguente frammento di codice Matlab/Octave (somma cumulativa):

```
v = [ 1 2 3 9 6 4 7 12 9 ]
n = length( v )
s = zeros( n );
i = 1;
while i <= n
    j = 1;
    while j <= i
        s( i ) = s( i ) + v( j );
        j = j + 1;
    end
    fprintf('Somma parziale %d indice %d \n', s( i ), j);
    i = i + 1;
end
```

Eseguendo il codice cosa appare a video?

v =

1 2 3 9 6 4 7 12 9

n =

9

Somma parziale 1 indice 2

Somma parziale 3 indice 3

Somma parziale 6 indice 4

Somma parziale 15 indice 5

Somma parziale 21 indice 6

Somma parziale 25 indice 7

Somma parziale 32 indice 8

Somma parziale 44 indice 9

Somma parziale 53 indice 10

Sessione estiva AA 2015/16 secondo appello

(1) Descrivere il modello di von Neumann.

(2) Cosa si intende per "fasi di esecuzione" di un programma?

- Un programmatore ha codificato un algoritmo che calcola minimo e massimo di un vettore, ma ha commesso degli errori grammaticali/sintattici. A parte la mancanza di commenti, quali sono gli errori?

```
1.  clc;
2.  clear;
3.
4.  A = floor( rand( 1..10 ) * 100 )
5.  val = [ A( 1 )  A( 1 )  ];
6.  pos = [ 1 1 ];
7.  for i = 1::length( A )
8.       if A( i ) < val( 1 )
9.           val( 1 ) = A( i );
10.          pos( 1 ) = i;
11.      end
12.      if A( i ) > val( 2 )
13.          val( 2 ) = A( i );
14.          pos( 2 ) = i;
15.      end
16.  end
17.
18. fprintf( 'Minimo = %d a %d\n', val( 1 ),  pos( 1 )  );
19. fprintf( 'Massimo = %d a %d\n', val( 2 ),  pos( 2 )  );
```

Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando cicli for/while: dato un vettore "V" di dimensione "N" costruire una matrice "M" che abbia gli elementi del vettore come diagonale e tutti i restanti pari a zero.

```
1. clear
2. clc
3. V=input(['digitare V']);
4. for i=1:length(V)
5.   for j=1:length(V)
6.     if i==j
7.       M(i,j)=V(i);
8.     else
9.       M(i,j)=0;
10.     end;
11.   end;
12. end;
13. disp(M)
```

- Un programmatore ha codificato un semplice algoritmo che implementa la funzione “linspace”. L’algoritmo è implementato correttamente ma il programmatore ha commesso degli errori sintattici/grammaticali. Quali?

```
1. %questo script calcola la function linspace
2. %valori di input
3. Xmin = 3;
4. Xmax = 15;
5. N = 10;
6.
7. %inizializzazioni (rapporto tra ampiezza intervallo e
8. % numero di punti)
9. k = ( Xmax - Xmin ) \ ( N - 1 );
10.i = 1;
11.v( i ) = Xmin;
12.
13.%loop che crea il vettore
14.for i == 2 : N
15.   v( i ) = v( i - 1 ) + k;
16.end
```

- Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando cicli for/while: creare una function che calcola il checksum delle due diagonali principali di una matrice NxN con N >=2; la matrice è passata come parametro d'ingresso, i parametri di uscita sono i due checksum delle diagonali principali. Nota: si consideri il checksum come la somma algebrica degli elementi della diagonale.

```
1.  function [ cs1 cs2 ] = Esame2017 ( M )
2.
3.  [ r c ] = size( M );
4.
5.  cs1 = 0;
6.  cs2 = 0;
7.
8.  if ( r ~= c )
9.    disp('Matrice non quadrata.');
10.   return;
11. end
12.
13. if ( r < 2 )
14.   disp('Matrice troppo piccola');
15.   return;
16. end
17.
18.
19. %primo e secondo checksum
20. for i = 1:r
21.     cs1 = cs1 + M( i, i );
22.     cs2 = cs2 + M( i, r - i + 1 );
23. end
24.
25. end
```

«if» più efficace

```
if ( (r ~= c) || ( r < 2 ) || ( c < 2 ) )
  disp('Matrice non quadrata o troppo piccola.');
  return;
end
```

(1) Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando cicli for/while: creare una function che effettui la concatenazione tra due vettori di dimensione N ed M alternando le componenti. N ed M possono assumere qualsiasi valore intero positivo maggiore di 1. Il vettore ottenuto sarà pertanto di dimensioni N+M, e verrà restituito come parametro d'uscita.

```
1.  function [ V ] = Concat( V1, V2 )
2.
3.  d1 = size( V1,2 );
4.  d2 = size( V2,2 );
5.
6.  %definisce ed azzera vettore di uscita
7.  for i = 1 : d1 + d2
8.    V( i ) = 0;
9.  end
10.
11. %V = zeros(d1+d2);
12.
13. Iv1 = 0;
14. Iv2 = 0;

15. % Esegue la concatenazione
16. for i = 1 : (d1 + d2)
17.   if mod(i,2) == 0 %indice pari
18.      Iv1 = Iv1 + 1;
19.      if Iv1<=d1
20.        V( i ) = V1( Iv1 );
21.      end
22.      if ( Iv1>d1 ) && ( i <= d1 + d2)
23.        Iv2 = Iv2 + 1;
24.        V( i ) = V2( Iv2 );
25.      end
26.   else              %indice dispari
27.      Iv2 = Iv2 + 1;
28.      if Iv2<=d2
29.        V( i ) = V2( Iv2 );
30.      end
31.      if ( Iv2>d2 ) && ( i <= d1 + d2)
32.         Iv1 = Iv1 + 1;
33.         V( i ) = V1( Iv1 );
34.      end
35.   end
36. end
37.
```

(1) Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando cicli for/while: creare una function che abbia come parametro di ingresso una matrice A di dimensioni NxM e come parametro di uscita una matrice B di dimensioni Mx2 che contiene nella prima colonna prodotto dei valori delle M colonne di A e nella seconda la somma dei medesimi valori.

(1) Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando solo cicli *for*/*while*, istruzioni condizionali (*if*, *switch*), e se ritenuto necessario le funzioni *zeros e size*: creare una *function* che riceva in ingresso due vettori V1 e V2 di dimensione N (controllare che la condizione sia verificata) e restituisca una matrice NxN che ha una diagonale composta dai valori di V1 sommati ai valori di V2 divisi per 2 (ma solo per i valori di V2 maggiori di zero).

```
1. function [ M ] = Esame2018 ( V1, V2 )
2.
3. r1 = length( V1 );
4. r2 = length( V2 );
5.
6. M = zeros( r1 );
7.
8. if ( r1 ~= r2 )
9.   disp('Vettori con dimensioni incompatibili');
10.  return;
11.end
12.
13.for i = 1:r1
14.    M( i, i ) = V1( i );
15.    if  V2( i ) > 0
16.      M( i, i ) = M( i, i ) + V2( i ) / 2
17.    end
18.end
19.
20.end
```

- Si chiede di proporre una soluzione Matlab/Octave al seguente problema, usando solo cicli *for*/*while*, istruzioni condizionali (*if*, *switch*), e se ritenuto necessario le funzioni *zeros e size*: creare una *function* che data in ingresso una matrice R*C con R e C >1 (verificare la condizione e prevedere una configurazione di uscita che segnali l'errore) ritorni una matrice di dimensioni R*(C+1). Essa conterrà il valore 1 quando il corrispondente valore della matrice d'ingresso è positivo, 99 se nullo e -1 se negativo. Nella colonna C+1 riportare la media dei valori della riga (della matrice d'ingresso).

```
1.  function UM = Esame2019( M )
2.
3.      [ R C ] = size( M )
4.      UM = zeros( R, C + 1 );
5.      i = 1;
6.      while i <= R
7.          j = 1;
8.          CS = 0;
9.          while j <= C
10.             Val = M( i, j );
11.             if (Val < 0)
12.                 UM( i, j ) = -1;
13.             elseif ( Val == 0 )
14.                 UM( i, j ) = 99;
15.             elseif ( Val > 0 )
16.                 UM( i, j ) = 1;
17.             end
18.             CS = CS + Val;
19.             j = j + 1;
20.         end
21.         CS = CS/C;
22.         % Qui J è pari a C+1
23.         UM( i, j ) = CS;
24.         i = i + 1;
25.     end
26.
27.
28. end
```

Codice

- Qualche esempio di codice e qualche soluzione degli esercizi assegnati nei moduli precedenti.
- Diamo esempi di stile

Applicazione del teorema di Bolzano (teorema degli zeri)

Teorema degli zeri: data una funzione continua in un certo intervallo chiuso e finito [a,b], se sussiste la condizione per cui la funzione assume segno opposto agli estremi dell'intervallo, allora esiste almeno un x0 dove la funzione si annulla.

f continua in [a,b] , f(a)*f(b)<0 >> esiste almeno un x0 tale che f(x0)=0

Esercizio:

Scrivere una funzione che riceva in input i valori estremanti di un intervallo [a,b] chiuso tale che f(a)*f(b)<0 e ritorni in output il valore per cui la funzione si annulla.

Algoritmo di bisezione: dividere in 2 parti uguali l'intervallo [a,b] e considerare il semintervallo ove la funzione assume agli estremi valori di segno discordi e reiteriamo su quell'intervallo.

Soluzione:

```
function [c, valido] = Teorema_Degli_Zeri(a,b,toll)
   % applicazione del teorema degli zeri (Bolzano), riceve in input [a,b] e la tolleranza come condizione di uscita
   % inizializza le variabili di uscita
   valido = false;
   c = 0;
   % crea un vettore x di punti equidistanti all'interno di [a,b] per tracciare la funzione
   x = linspace(a,b,100);
   y = myFunction(x);
   plot(x,y);
   grid on;
   if (a >= b)  % controlla se l'intervallo a,b è valido
      fprintf("intervallo non valido b deve essere maggiore di a \n")
   elseif (myFunction(a)*myFunction(b)>0)
      fprintf("intervallo non valido f(b) deve essere di segno discorde da f(a) \n")
   elseif abs(b-a)<=toll
      fprintf("la tolleranza deve essere minore di b-a \n")
   else
      valido = true;
      while abs(b-a)>toll
        c = (b+a)/2;
        if (myFunction(c)*myFunction(a)>0)
          a = c;
        else
          b = c;
        end
      end
   end
end

function y=myFunction(x)
  y=10*x+15;
end
```

Homework

Considerate un vettore riga V di lunghezza N. progettare ed implementare:

1. Una function che ritorni il minimo di V ed il suo indice. La funzione prende come input V.
2. Una function che trova un valore x in V e se trovato ne ritorna il relativo indice (vietato usare find!). La funzione prende come input V e x.
3. Una function che calcola la media degli elementi in V. La funzione prende come ingresso V.
4. Una function che ritorna il minimo e il massimo degli elementi di V e relativi indici. La funzione prende come ingresso V.
5. Una function che calcola la media degli elementi contenuti in V e ritorna l'indice del valore più vicino alla media. La funzione prende come ingresso V.

Considerate poi una matrice A di MxN

1. Risolvere i problemi di cui sopra (1-5) per la matrice A (al posto del vettore V)

Soluzione 1

- Una function che ritorna il minimo di V ed il suo indice. La funzione prende come input V

```
function [ min index ] = myMin( A )
%myMin returns the minimum value in vector A and its index

min = A(1);
index = 1;

for i=1:length(A)

    if A(i) < min
        min = A(i);
        index = i;

    end

end

end
```

Break

- L'istruzione break è molto utilizzata
- Consente di «abortire» prematuramente un loop
- Se i loop sono «innestati» abortisce solo quello corrente, per uscire da più di un loop si deve «propagare» il comando tramite variabili intermedie
- Al di fuori di un loop si deve usare return
- Per esempio:

```
i = 0;
while i < 5
      i = i + 1;
        if (i > 2)
            break;
        end
end
```

Soluzione 2

- Una function (oppure uno script) t che trova un valore x in V e se trovato ne ritorna il relativo indice (non usare find!). La funzione prende come input V e x.

```
A = [ 4 7 8 9 12 2 4 6];
 theValue=12;
 for index = 1:length(A)
     if A(index) == theValue
         break;
     end
 end

 %% Ora dobbiamo controllare se A(i) == theValue oppure se il valore non esiste
 %% Tre possibilità complessive:
 %% i<N,
 %% i==length(A) and A(i)~=theValue
 %% i==length(A) and A(i)==theValue

 if index==length(A) && A(index)~=theValue
         fprintf('Valore non trovato \n');
 else
     fprintf('Il valore %d si trova in posizione %d in A \n', theValue,index);
 end
```

Trovare il valore in una matrice: script

```
A = [ 4 7 8 9 12 2 4 6;
      5 7 8 12 2 45 56 0];
 theValue=9;
 n=size(A,1);
 m=size(A,2); %si può fare in altro modo?
 esci = false;
 for i = 1:n
     for j = 1:m
        if A( i, j ) == theValue
            esci = true;
            break;
        end
     end
     if esci
        break;
     end
 end
 fprintf('Il valore %d si trova nella posizione
(%d,%d) in A \n', theValue, i, j);
```

```
function [idx1 idx2]=find_a_value_in_M(A,theValue)
% Trova il valore "theValue" nella matrice A e ritorna
% l'indice
n = size(A,1);
m = size(A,2);
esci = false;
idx1 = 0;
idx2 = 0;
for i = 1:n
     for j = 1:m
        if A( i, j ) == theValue
            esci=true;
            idx1=i;
            idx2=j;
            break;
        end
     end
     if esci
        break;
     end
end
```

Soluzione 3

- Una function che calcola la media degli elementi in V. La funzione prende come ingresso V.

```
A = [4 6 7 10 19];

function [ AVG ] = Average( V )
% Calcolo della media
% primo metodo

a = 0;

for x = V
    a = a + x;
end

AVG = a / length(V)

end
```

Soluzione 5

Una function che calcola la media degli elementi contenuti in V e ritorna l'indice del valore più vicino alla media. La funzione prende come ingresso V.

- E' come trovare il minimo di un vettore
  - perchè?
- Definiamo la distanza tra due punti x1 e x2
  - d=|x2 - x1|
- Dobbiamo trovare l'elemento del vettore che ha la minima distanza dal valor medio

Soluzione

se A è l'ingresso i passi sono:

1. avg = Average(A); % calcolo valor medio
2. diff=abs(A - avg); % calcolo «vettore delle distanze»
3. x = myMin(diff); % la posizione voluta

Vogliamo una funzione che ha un comportamento definito nelle seguenti condizioni:

- A=[ ] (in questo esempio consideriamo solo il caso semplice del vettore riga o colonna)
- size(A) = (1,n) || (n,1)
- size(A) = (n,m)

Dunque, se consideriamo anche il caso delle matrici :

sia myAvg che myMin devono avere un comportamento definito date le condizioni elencate sopra : homework, fatelo come esercizio

Codifica

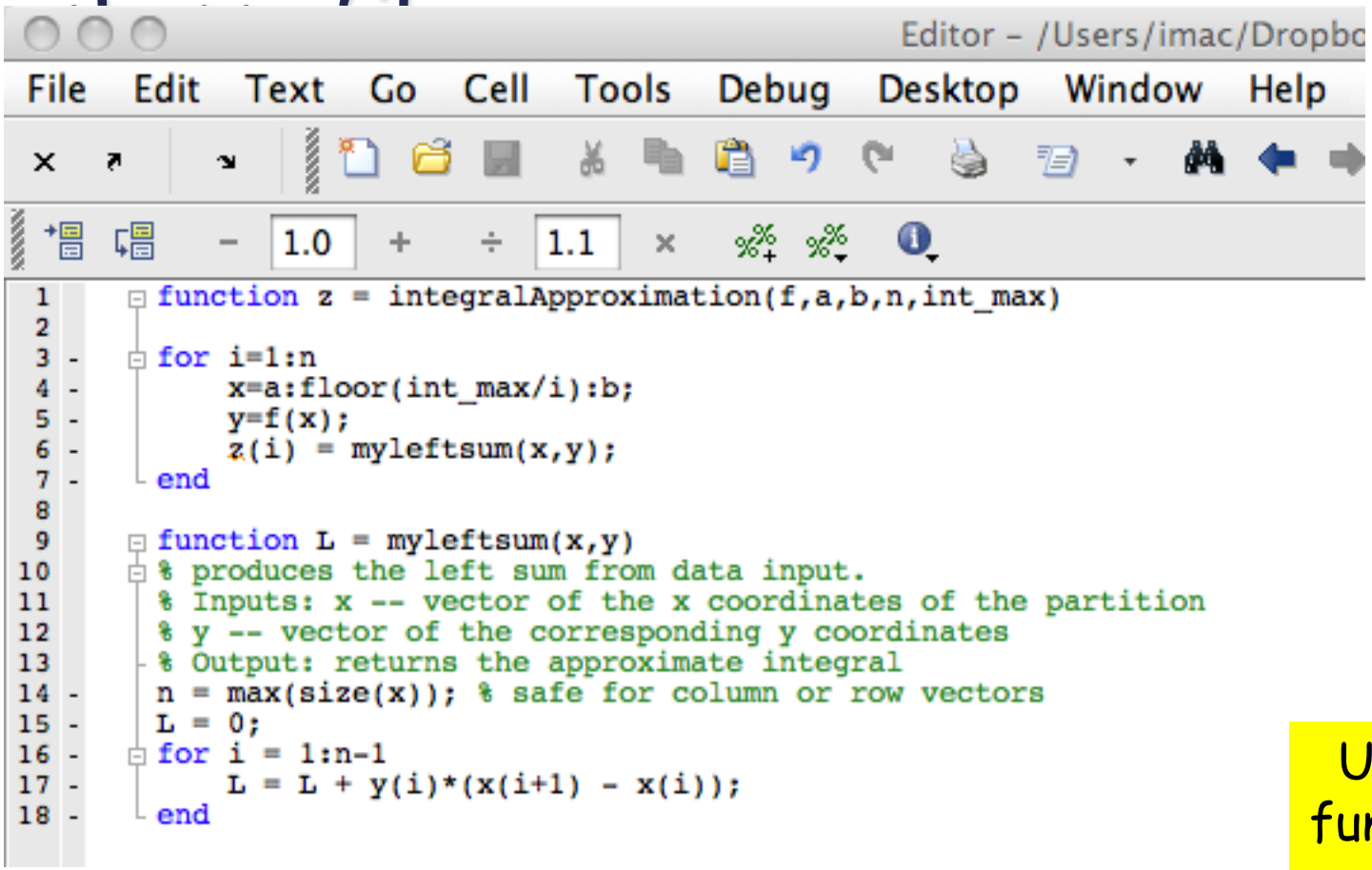
```
function [ avg, x, pos ] = closestToAvg( V )


  if isempty(V) || ndims(V)>1 % controlla se V è
    % oppure se il numero delle dimensioni di V è
        avg=0; x=0; pos=0;
      return;
  else % I parametri sono OK
      avg = Average(V);
      diff = abs(V-avg);
      [~,pos] = myMin(diff);
      x=V(pos);
  end

end
```

Problema: trovare un insieme di valori in un vettore

- Dato un vettore ed un insieme di valori, scrivere una funzione che restituisce “true” se un valore dell’insieme è nel vettore, “false” in caso contrario
  - Ingresso: vettore (A) e l’insieme di valori (pivot)
    - L’insieme pivot è rappresentato a sua volta da un vettore
  - Come definire l’uscita (il problema è stato ben formulato)?
    - soluzione: un vettore r della stessa taglia del pivot.
    - r(i)=true se pivot(i) è in A. Altrimenti r(i)=false.
  - Come gestire e segnalare input sbagliati
    - se r=[] vuol dire che la funzione abortisce

```
function r=findSet(A,pivot)
% Trova le occorrenze di pivot in A
% ossia gli elementi di A contenenti i valori
% Input: A -- un vettore
% pivot - un vettore contenente gli elementi da cercare in A
% Output: r - un vettore di booleani della stessa taglia del vettore  dei  %
pivot. r(i)=true se
%pivot(i) è contenuto in A. Altrimenti r(i)=false. if r=[] vuol dire che la
function abortisce
%
n_a=length(A);
n_p=length(pivot);
if n_a<n_p
    r=[];
    exit;
end
r(1:n_p)=false;
for i = 1:n_a
     for j = 1:n_p
        if A(i) == pivot(j)
           r(j)=true;
           break;
        end
     end
end
```

Invece di due funzioni separate si possono definire due funzioni nello



Utile per funzioni "di servizio"

Codice

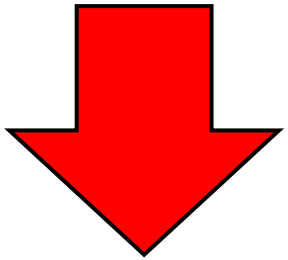
- Qualche ulteriore esempio di codice
- …ancora, approfittiamo per definire delle regole di buona programmazione

Problema

- Dato un vettore V of lunghezza L trasformarlo in una matrice NxM, dove L=NxM

idx

1	2	3	4	5	6
V(1)	V(2)	V(3)	V(4)	V(5)	V(6)



idx deve essere tradotto in un indice

(row, col)

(row, col)	1	2
1	A(1,1)=V(1)	A(1,2)=V(4)
2	A(2,1)=V(2)	A(2,2)=V(5)
3	A(3,1)=V(3)	A(3,2)=V(6)

Solution

- Da un indice “unidimensionale” (idx) a “bidimensionale” (row, col) di una matrice NxM

$$row = \begin{cases} N & \text{if } (idx \bmod N) == 0 \\ k = idx \bmod N & \text{otherwise} \end{cases}$$

$$col = \begin{cases} \lfloor idx / N \rfloor & \text{if } (idx \bmod M) == 0 \\ \lfloor idx / N \rfloor + 1 & \text{otherwise} \end{cases}$$

Codice

```
function A=vectorToMatrix(V,n,m)
%questa function ridimensiona un vettore V di taglia (1,n*m)
%in una matrice A di taglia n*m
%INPUT: un vettore  V e due  scalari n ed m
%OUTPUT la matrice A

%controlla se n ed m sono interi e…
%...controlla se V è vuoto o se la dimensione di V non si accorda
con n ed m
if floor(n)~=n || floor(m)~=m || isempty(V) ||
                    length(V) ~= n*m
    A=[];
```

DOBBIAMO controllare se i parametri d'ingresso sono corretti e DOBBIAMO definire il comportamento della funzione per ingressi sbagliati

Codice

```
function A=vectorToMatrix(V,n,m)
%questa function ridimensiona un vettore V di taglia (1,n*m)
%in una matrice A di taglia n*m
%INPUT: un vettore  V e due  scalari n ed m
%OUTPUT la matrice A

%controlla se n ed m sono interi e…
%...controlla se V è vuoto o se la dimensione di V non si accorda
con n ed m
if floor(n)~=n || floor(m)~=m || isempty(V) ||
                    length(V) ~= n*m
    A=[];
    return;
end
```

```
function A=vectorToMatrix(V,n,m)
%questa function ridimensiona un vettore V di taglia (1,n*m)
%in una matrice A di taglia n*m
%INPUT: un vettore  V e due  scalari n ed m
%OUTPUT la matrice A

%controlla se n ed m sono interi e…
%...controlla s V è vuoto o se la dimensione di V non si accorda con n ed
m
if floor(n)~=n || floor(m)~=m || isempty(V) ||
        length(V) ~= n*m || ndims(V)>2
    A=[];
    return;
end

for i=1:n*m
     row=mod(i,n);
     if row==0
         row=n;
         col=floor(i./n);
     else
         col=floor(i./n)+1;
     end
     A(row,col)=V(i);
end
```

Homework

- Problema: data una matrice A di taglia NxM trasformarla in un vettore di taglia L=NxM